**入门训练 Fibonacci数列**

问题描述

Fibonacci数列的递推公式为：Fn=Fn-1+Fn-2，其中F1=F2=1。

当n比较大时，Fn也非常大，现在我们想知道，Fn除以10007的余数是多少。

输入格式

输入包含一个整数n。

输出格式

输出一行，包含一个整数，表示Fn除以10007的余数。

说明：在本题中，答案是要求Fn除以10007的余数，因此我们只要能算出这个余数即可，而不需要先计算出Fn的准确值，再将计算的结果除以10007取余数，直接计算余数往往比先算出原数再取余简单。

样例输入

10

样例输出

55

样例输入

22

样例输出

7704

数据规模与约定

1 <= n <= 1,000,000。

本题的C++参考代码如下：

#include <stdlib.h>

#include <stdio.h>

#define MOD 10007

#define MAXN 1000001

int n, i, F[MAXN];

int main()

{

scanf("%d", &n);

F[1] = 1;

F[2] = 1;

for (i = 3; i <= n; ++i)

F[i] = (F[i-1] + F[i-2]) % MOD;

printf("%d\n", F[n]);

return 0;

}

本题的C参考代码如下：

F[MAXN];

int main()

{

scanf("%d", &n);

F[1] = 1;

F[2] = 1;

for (i = 3; i <= n; ++i)

F[i] = (F[i-1] + F[i-2]) % MOD;

printf("%d\n", F[n]);

return 0;

}

**入门训练 圆的面积**

问题描述

给定圆的半径r，求圆的面积。

输入格式

输入包含一个整数r，表示圆的半径。

输出格式

输出一行，包含一个实数，四舍五入保留小数点后7位，表示圆的面积。

说明：在本题中，输入是一个整数，但是输出是一个实数。

对于实数输出的问题，请一定看清楚实数输出的要求，比如本题中要求保留小数点后7位，则你的程序必须**严格的**输出7位小数，输出过多或者过少的小数位数都是不行的，都会被认为错误。

实数输出的问题如果没有特别说明，舍入都是按四舍五入进行。

样例输入

4

样例输出

50.2654825

数据规模与约定

1 <= r <= 10000。

提示

本题对精度要求较高，请注意π的值应该取较精确的值。你可以使用常量来表示π，比如PI=3.14159265358979323，也可以使用数学公式来求π，比如PI=atan(1.0)\*4。

本题的C++参考代码如下：

#include <stdio.h>

#include <math.h>

int main()

{

int r;

double s, PI;

scanf("%d", &r);

PI = atan(1.0) \* 4;

s = PI \* r \* r;

printf("%.7lf", s);

return 0;

}

本题的C参考代码如下:

#include <stdio.h>

#include <math.h>

int main()

{

int r;

double s, PI;

scanf("%d", &r);

PI = atan(1.0) \* 4;

s = PI \* r \* r;

printf("%.7lf", s);

return 0;

}

**入门训练 序列求和**

问题描述

求1+2+3+...+n的值。

输入格式

输入包括一个整数n。

输出格式

输出一行，包括一个整数，表示1+2+3+...+n的值。

样例输入

4

样例输出

10

样例输入

100

说明：有一些试题会给出多组样例输入输出以帮助你更好的做题。

一般在提交之前所有这些样例都需要测试通过才行，但这不代表这几组样例数据都正确了你的程序就是完全正确的，潜在的错误可能仍然导致你的得分较低。

样例输出

5050

数据规模与约定

1 <= n <= 1,000,000,000。

说明：请注意这里的数据规模。

本题直接的想法是直接使用一个循环来累加，然而，当数据规模很大时，这种“暴力”的方法往往会导致超时。此时你需要想想其他方法。你可以试一试，如果使用1000000000作为你的程序的输入，你的程序是不是能在规定的上面规定的时限内运行出来。

本题另一个要值得注意的地方是答案的大小不在你的语言默认的整型(int)范围内，如果使用整型来保存结果，会导致结果错误。

如果你使用C++或C语言而且准备使用printf输出结果，则你的格式字符串应该写成%I64d以输出long long类型的整数。

本题的C++参考代码如下：

#include <iostream>

using namespace std;

int main()

{

long long n;

cin >> n;

cout << (1+n) \* n / 2;

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

int main()

{ tmp的作用是什么？

int n;

long long ret = 0, tmp;

scanf("%d", &n);

tmp = n;

printf("%I64d", (1+tmp) \* tmp / 2);

return 0;

}

**入门训练 A+B问题**

问题描述

输入A、B，输出A+B。

说明：在“问题描述”这部分，会给出试题的意思，以及所要求的目标。

输入格式

输入的第一行包括两个整数，由空格分隔，分别表示A、B。

说明：“输入格式”是描述在测试你的程序时，所给的输入一定满足的格式。

做题时你应该假设所给的输入是一定满足输入格式的要求的，所以你不需要对输入的格式进行检查。多余的格式检查可能会适得其反，使用你的程序错误。

在测试的时候，系统会自动将输入数据输入到你的程序中，你不能给任何提示。比如，你在输入的时候提示“请输入A、B”之类的话是不需要的，这些多余的输出会使得你的程序被判定为错误。

输出格式

输出一行，包括一个整数，表示A+B的值。

说明：“输出格式”是要求你的程序在输出结果的时候必须满足的格式。

在输出时，你的程序必须满足这个格式的要求，不能少任何内容，也不能多任何内容。如果你的内容和输出格式要求的不一样，你的程序会被判断为错误，包括你输出了提示信息、中间调试信息、计时或者统计的信息等。

样例输入

12 45

说明：“样例输入”给出了一组满足“输入格式”要求的输入的例子。

这里给出的输入只是可能用来测试你的程序的一个输入，在测试的时候，还会有更多的输入用来测试你的程序。

样例输出

57

说明：“样例输出”给出了一组满足“输出格式”要求的输出的例子。

样例输出中的结果是和样例输入中的是对应的，因此，你可以使用样例的输入输出简单的检查你的程序。

要特别指出的是，能够通过样例输入输出的程序并不一定是正确的程序，在测试的时候，会用很多组数据进行测试，而不局限于样例数据。有可能一个程序通过了样例数据，但测试的时候仍只能得0分，可能因为这个程序只在一些类似样例的特例中正确，而不具有通用性，再测试更多数据时会出现错误。

比如，对于本题，如果你写一个程序不管输入是什么都输入57，则样例数据是对的，但是测试其他数据，哪怕输入是1和2，这个程序也输出57，则对于其他数据这个程序都不正确。

数据规模与约定

-10000 <= A, B <= 10000。

说明：“数据规模与约定”中给出了试题中主要参数的范围。

这个范围对于解题非常重要，不同的数据范围会导致试题需要使用不同的解法来解决。比如本题中给的A、B范围不大，可以使用整型(int)来保存，如果范围更大，超过int的范围，则要考虑其他方法来保存大数。

有一些范围在方便的时候是在“问题描述”中直接给的，所以在做题时不仅要看这个范围，还要注意问题描述

本题的C参考代码如下：

#include <stdio.h>

int main()

{

int a, b;

scanf("%d%d", &a, &b);

printf("%d", a+b);

return 0;

}

本题的C++参考代码如下：

#include <iostream>

using namespace std;

int main()

{

int a, b;

cin >> a >> b;

cout << a + b;

return 0;

}

**基础练习 数列排序**

问题描述

　　给定一个长度为n的数列，将这个数列按从小到大的顺序排列。1<=n<=200

输入格式

　　第一行为一个整数n。  
　　第二行包含n个整数，为待排序的数，每个整数的绝对值小于10000。

输出格式

　　输出一行，按从小到大的顺序输出排序后的数列。

样例输入

5  
8 3 6 4 9

样例输出

3 4 6 8 9

本题的C++参考代码如下：

#include<iostream>

#include<algorithm>

using namespace std;

i

nt cmp(int a,int b)

{

return a<b;

}

int main()

{

int n;

while(cin>>n)

{

int a[205];

for(int i=0;i<n;i++)

{

cin>>a[i];

}

sort(a,a+n,cmp);

cout<<a[0];

for(int i=1;i<n;i++)

{

cout<<' '<<a[i];

}

cout<<endl;

}

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

#include <stdlib.h>

void Print(int \*data,int n)

{

int i;

for(i=0;i<n;i++)

{

printf("%d ",data[i]);

}

printf("\n");

}

int Split(int \*data,int pre,int rear)

{

int value=data[pre];

while(pre<rear)

{

while(data[rear]>=value && pre<rear) rear--;

data[pre]=data[rear];

while(data[pre]<value && pre<rear) pre++;

data[rear]=data[pre];

}

data[pre]=value;

return pre;

}

//快速排序

void QuickSort(int \*data,int pre,int rear)

{

if(pre<rear)

{

int mid=Split(data,pre,rear);

QuickSort(data,pre,mid-1);

QuickSort(data,mid+1,rear);

}

}

int main()

{

int i;

int n;

int \*data;

scanf("%d",&n);

data=(int \*)malloc(sizeof(int)\*n);

for(i=0;i<n;i++)

{

scanf("%d",&data[i]);

}

QuickSort(data,0,n-1);

Print(data,n);

return 0;

}

**基础练习 十六进制转八进制**

**问题描述**  
　　给定n个十六进制正整数，输出它们对应的八进制数。  
  
**输入格式**  
　　输入的第一行为一个正整数n （1<=n<=10）。  
　　接下来n行，每行一个由0~9、大写字母A~F组成的字符串，表示要转换的十六进制正整数，每个十六进制数长度不超过100000。  
  
**输出格式**  
　　输出n行，每行为输入对应的八进制正整数。  
  
　　**【注意**】  
　　输入的十六进制数不会有前导0，比如012A。  
　　输出的八进制数也不能有前导0。  
  
**样例输入**  
　　2  
　　39  
　　123ABC  
  
**样例输出**  
　　71  
　　4435274  
  
　　**【提示**】  
　　先将十六进制数转换成某进制数，再由某进制数转换成八进制。

本题的C++参考代码如下：

#include <iostream>

#include <stdio.h>

#include <string.h>

#include <STDLIB.H>

/\* run this program using the console pauser or add your own getch, system("pause") or input loop \*/

int GetI(char c)

{

return c>>4&1?c&15:(c&15)+9;

}

int main(int argc, char \*argv[]) {

char arr[200001] = {'\0'};

char brr[400001] = {'\0'};

int n = 0;

int i = 0;

scanf("%d",&n);

for(i = 0;i < n;i++)

{

scanf("%s",arr);

int m[3] = {1,16,256};

int len = strlen(arr);

int j = len-1;

int a,b,c;

a = b = c = 0;

int k = 0,l = 0;

int count = 0;

while(j>-1)

{

a += (arr[j]>>4&1?arr[j]&15:(arr[j]&15)+9)\*m[k]; //个位

if(k==2||j==0)

{

while(a)

{

brr[l++] = ((a&7)|48);

a = a>>3;

count++;

}

while(j!=0&&count<4)

{

brr[l++] = '0';

count++;

}

count = 0;

}

k = (k+1)%3;

j--;

}

strrev(brr);

printf("%s\n",brr);

memset(arr,'\0',(sizeof(char)\*200001));

memset(brr,'\0',(sizeof(char)\*400001));

}

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MaxSize 100000

void saveB(char \*b,char c3,char c2,char c1,char c0,int start)

{

b[start]=c3;

b[start+1]=c2;

b[start+2]=c1;

b[start+3]=c0;

}

int htob(char \*h,char \*b)

{

int i,j;

int hl=strlen(h);

for(i=0;i<hl;i++)

switch(h[i])

{

case '0':

{

saveB(b,'0','0','0','0',4\*i);

break;

}

case '1':

{

saveB(b,'0','0','0','1',4\*i);

break;

}

case '2':

{

saveB(b,'0','0','1','0',4\*i);

break;

}

case '3':

{

saveB(b,'0','0','1','1',4\*i);

break;

}

case '4':

{

saveB(b,'0','1','0','0',4\*i);

break;

}

case '5':

{

saveB(b,'0','1','0','1',4\*i);

break;

}

case '6':

{

saveB(b,'0','1','1','0',4\*i);

break;

}

case '7':

{

saveB(b,'0','1','1','1',4\*i);

break;

}

case '8':

{

saveB(b,'1','0','0','0',4\*i);

break;

}

case '9':

{

saveB(b,'1','0','0','1',4\*i);

break;

}

case 'A':

{

saveB(b,'1','0','1','0',4\*i);

break;

}

case 'B':

{

saveB(b,'1','0','1','1',4\*i);

break;

}

case 'C':

{

saveB(b,'1','1','0','0',4\*i);

break;

}

case 'D':

{

saveB(b,'1','1','0','1',4\*i);

break;

}

case 'E':

{

saveB(b,'1','1','1','0',4\*i);

break;

}

case 'F':

{

saveB(b,'1','1','1','1',4\*i);

break;

}

}

return 4\*hl;

}

int btoo(char \*b,char \*o,int bl)

{

int i,j;

int ol;

int value;

if(bl%3==0)

ol=bl/3;

else

ol=bl/3+1;

j=bl-1;

for(i=ol-1;i>=0;i--)

{

if(i>0)

o[i]=b[j]-48+(b[j-1]-48)\*2+(b[j-2]-48)\*4+48;

else

{

switch(j)

{

case 2:

o[i]=b[j]-48+(b[j-1]-48)\*2+(b[j-2]-48)\*4+48;

break;

case 1:

o[i]=b[j]-48+(b[j-1]-48)\*2+48;

break;

case 0:

o[i]=b[j];

break;

}

}

j=j-3;

}

return ol;

}

void printO(char \*o,int ol)

{

int i=0;

if(o[0]=='0')

i=1;

for(;i<ol;i++)

{

printf("%c",o[i]);

}

printf("\n");

}

main()

{

char h[MaxSize];

char b[4\*MaxSize];

char o[4\*MaxSize/3+1];

int n,i,bl,ol;

scanf("%d",&n);

getchar();

for(i=0;i<n;i++)

{

gets(h);

bl=htob(h,b);

ol=btoo(b,o,bl);

printO(o,ol);

}

}

**基础练习 十六进制转十进制**

问题描述

　　从键盘输入一个不超过8位的正的十六进制数字符串，将它转换为正的十进制数后输出。  
　　注：十六进制数中的10~15分别用大写的英文字母A、B、C、D、E、F表示。

样例输入

FFFF

样例输出

65535

本题的C++参考代码如下：

#include<iostream>

#include<string>

using namespace std;

int main()

{

string s;

while(cin>>s)

{

int leth=s.length();

long long sum=0;

for(int i=0;i<leth;i++)

{

if(s[i]>='A'&&s[i]<='F')

{

sum=sum\*16+s[i]-'A'+10;

// cout<<sum<<endl;

}

else

{

sum=sum\*16+s[i]-'0';

//cout<<sum<<endl;

}

}

cout<<sum<<endl;

}

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

#include<string.h>

#include<math.h>

int main()

{

double sum=0,x;

char a[8];

int len,i=0;

gets(a);

len=strlen(a);

while(len)

{

if(a[len-1]>='A'&&a[len-1]<='F')

x=(a[len-1]-'7')\*pow(16,i++);

else

x=(a[len-1]-'0')\*pow(16,i++);

sum+=x;

len--;

}

printf("%.0lf",sum);

return 0;

}

**基础练习 十进制转十六进制**

问题描述

　　十六进制数是在程序设计时经常要使用到的一种整数的表示方式。它有0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F共16个符号，分别表示十进制数的0至15。十六进制的计数方法是满16进1，所以十进制数16在十六进制中是10，而十进制的17在十六进制中是11，以此类推，十进制的30在十六进制中是1E。  
　　给出一个非负整数，将它表示成十六进制的形式。

输入格式

　　输入包含一个非负整数a，表示要转换的数。0<=a<=2147483647

输出格式

　　输出这个整数的16进制表示

样例输入

30

样例输出

1E

本题的C++参考代码如下：

#include<iostream>

#include<cstdio>

using namespace std;

int main()

{

int n;

char s[100000];

while(cin>>n)

{

int k=0;

if(n==0)

{

cout<<0;

}

else

{

while(n!=0)

{

if(n%16>=10)

{

s[k++]='A'+n%16-10;

}

else

{

s[k++]='0'+n%16;

}

n=n/16;

}

for(int i=k-1;i>=0;i--)

{

cout<<s[i];

}

}

cout<<endl;

}

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

#include <stdlib.h>

char data[]={'0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'};

//输出

void Print(char \*num,int n)

{

int i;

for(i=n-1;i>=0;i--) printf("%c",num[i]);

printf("\0");

printf("\n");

}

//将十六进制数转换为十进制数

int Transform(char \*num,long long value)

{

int n=0;

while(value>=16)

{

num[n++]=data[value%16];

value/=16;

}

num[n++]=data[value%16];

return n;

}

int main()

{

long long value;

char num[10];

int n;

scanf("%I64d",&value);

n=Transform(num,value);

Print(num,n);

return 0;

}

**基础练习 十进制转十六进制**

问题描述

　　十六进制数是在程序设计时经常要使用到的一种整数的表示方式。它有0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F共16个符号，分别表示十进制数的0至15。十六进制的计数方法是满16进1，所以十进制数16在十六进制中是10，而十进制的17在十六进制中是11，以此类推，十进制的30在十六进制中是1E。  
　　给出一个非负整数，将它表示成十六进制的形式。

输入格式

　　输入包含一个非负整数a，表示要转换的数。0<=a<=2147483647

输出格式

　　输出这个整数的16进制表示

样例输入

30

样例输出

1E

本题的C++参考代码如下：

#include<iostream>

#include<cstdio>

using namespace std;

int main()

{

int n;

char s[100000];

while(cin>>n)

{

int k=0;

if(n==0)

{

cout<<0;

}

else

{

while(n!=0)

{

if(n%16>=10)

{

s[k++]='A'+n%16-10;

}

else

{

s[k++]='0'+n%16;

}

n=n/16;

}

for(int i=k-1;i>=0;i--)

{

cout<<s[i];

}

}

cout<<endl;

}

return 0;

}

本题的C参考代码如下:

#include <stdio.h>

#include <stdlib.h>

char data[]={'0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'};

//输出

void Print(char \*num,int n)

{

int i;

for(i=n-1;i>=0;i--) printf("%c",num[i]);

printf("\0");

printf("\n");

}

//将十六进制数转换为十进制数

int Transform(char \*num,long long value)

{

int n=0;

while(value>=16)

{

num[n++]=data[value%16];

value/=16;

}

num[n++]=data[value%16];

return n;

}

int main()

{

long long value;

char num[10];

int n;

scanf("%I64d",&value);

n=Transform(num,value);

Print(num,n);

return 0;

}

**基础练习 特殊回文数**

问题描述

　　123321是一个非常特殊的数，它从左边读和从右边读是一样的。  
　　输入一个正整数n， 编程求所有这样的五位和六位十进制数，满足各位数字之和等于n 。

输入格式

　　输入一行，包含一个正整数n。

输出格式

　　按从小到大的顺序输出满足条件的整数，每个整数占一行。

样例输入

52

样例输出

899998  
989989  
998899

数据规模和约定

　　1<=n<=54。

本题的C++参考代码如下：

#include <iostream>

using namespace std;

int main()

{

int n,a,b,c,t;

cin>>n;

for(a=1;a<10;a++)

for(b=0;b<10;b++)

for(c=0;c<10;c++)

{

t=a\*10001+b\*1010+c\*100;

if(2\*a+2\*b+c==n)

cout<<t<<endl;

}

for(a=1;a<10;a++)

for(b=0;b<10;b++)

for(c=0;c<10;c++)

{

t=a\*100001+b\*10010+c\*1100;

if(2\*a+2\*b+2\*c==n)

cout<<t<<endl;

}

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

int main()

{

int a,b,c,d,e,f,t,all;

scanf("%d",&t);

for(a=1;a<10;a++)

for(b=0;b<10;b++)

for(c=0;c<10;c++)

for(d=0;d<10;d++)

for(e=0;e<10;e++)

{

if(a==e)

if(b==d)

{

all=a+b+c+d+e;

if(all==t)

printf("%d\n",a\*10000+b\*1000+c\*100+d\*10+e);

}

}

for(a=1;a<10;a++)

for(b=0;b<10;b++)

for(c=0;c<10;c++)

for(d=0;d<10;d++)

for(e=0;e<10;e++)

for(f=0;f<10;f++)

{

if(a==f)

if(b==e)

if(c==d)

{

all=a+b+c+d+e+f;

if(all==t)

printf("%d\n",a\*100000+b\*10000+c\*1000+d\*100+e\*10+f);

}

}

return 0;

}

**基础练习 回文数**

问题描述

　　1221是一个非常特殊的数，它从左边读和从右边读是一样的，编程求所有这样的四位十进制数。

输出格式

　　按从小到大的顺序输出满足条件的四位十进制数。

本题的C++参考代码如下：

#include<stdio.h>

int main()

{

for(int i1=1;i1<10;i1++)

{

for(int i2=0;i2<10;i2++)

{

for(int i3=0;i3<10;i3++)

{

for(int i4=0;i4<10;i4++)

{

if(i1==i4 && i2==i3)

printf("%d%d%d%d\n",i1,i2,i3,i4);

}

}

}

}

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

int main()

{

int a,b,c,d,i;

for(i=1000;i<=9999;i++)

{

a=i/1000;

b=i/100%10;

c=i/10%10;

d=i%10;

if(a==d&&b==c)

printf("%d\n",i);

}

return 0;

}

**基础练习 特殊的数字**

问题描述

　　153是一个非常特殊的数，它等于它的每位数字的立方和，即153=1\*1\*1+5\*5\*5+3\*3\*3。编程求所有满足这种条件的三位十进制数。

输出格式

　　按从小到大的顺序输出满足条件的三位十进制数，每个数占一行。

本题的C++参考代码如下：

#include<iostream>

using namespace std;

int main()

{

int i,j,k;

for(i=1;i<=9;i++)

{

for(j=0;j<=9;j++)

{

for(k=0;k<=9;k++)

{

if(i\*100+j\*10+k==i\*i\*i+j\*j\*j+k\*k\*k)

{

cout<<i\*100+j\*10+k<<endl;

}

}

}

}

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

int main()

{

int i,j,k,n;

for(n=100;n<1000;n++)

{

i=n/100;

j=n/10%10;

k=n%10;

if(i\*100+j\*10+k==i\*i\*i+j\*j\*j+k\*k\*k)

printf("%d\n",n);

}

return 0;

}

**基础练习 杨辉三角形**

问题描述

杨辉三角形又称Pascal三角形，它的第i+1行是(a+b)i的展开式的系数。

它的一个重要性质是：三角形中的每个数字等于它两肩上的数字相加。

下面给出了杨辉三角形的前4行：

   1

  1 1

 1 2 1

1 3 3 1

给出n，输出它的前n行。

输入格式

输入包含一个数n。

输出格式

输出杨辉三角形的前n行。每一行从这一行的第一个数开始依次输出，中间使用一个空格分隔。请不要在前面输出多余的空格。

样例输入

4

样例输出

1  
1 1  
1 2 1  
1 3 3 1

数据规模与约定

1 <= n <= 34。

本题的C++参考代码如下：

int n;

int a[MAXN][MAXN];

int main()

{

cin >> n;

a[0][0] = 1;

for (int i = 0; i < n; ++i)

{

a[i][0] = a[i][i] = 1;

for (int j = 1; j < i; ++j)

a[i][j] = a[i-1][j-1] + a[i-1][j];

}

for (int i = 0; i < n; ++i)

{

for (int j = 0; j <= i; ++j)

cout << a[i][j] << " ";

cout << endl;

}

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

#define MAXN 40

int n;

int a[MAXN][MAXN];

int main()

{

int i, j;

scanf("%d", &n);

a[0][0] = 1;

for (i = 0; i < n; ++i)

{

a[i][0] = a[i][i] = 1;

for (j = 1; j < i; ++j)

a[i][j] = a[i-1][j-1] + a[i-1][j];

}

for (i = 0; i < n; ++i)

{

for (j = 0; j <= i; ++j)

printf("%d ", a[i][j]);

printf("\n");

}

return 0;

}

**基础练习 查找整数**

问题描述

给出一个包含n个整数的数列，问整数a在数列中的第一次出现是第几个。

输入格式

第一行包含一个整数n。

第二行包含n个非负整数，为给定的数列，数列中的每个数都不大于10000。

第三行包含一个整数a，为待查找的数。

输出格式

如果a在数列中出现了，输出它第一次出现的位置(位置从1开始编号)，否则输出-1。

样例输入

6  
1 9 4 8 3 9  
9

样例输出

2

数据规模与约定

1 <= n <= 1000。

本题的C++参考代码如下：

int n, a, ans;

int s[MAXN];

int main()

{

cin >> n;

for (int i = 0; i < n; ++i)

cin >> s[i];

cin >> a;

ans = -1;

for (int i = 0; i < n; ++i)

{

if (s[i] == a)

{

ans = i + 1;

break;

}

}

cout << ans << endl;

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

#define MAXN 10001

int n, a, ans;

int s[MAXN];

int main()

{

int i;

scanf("%d", &n);

for (i = 0; i < n; ++i)

scanf("%d", &s[i]);

scanf("%d", &a);

ans = -1;

for (i = 0; i < n; ++i)

{

if (s[i] == a)

{

ans = i + 1;

break;

}

}

printf("%d", ans);

return 0;

}

**基础练习 数列特征**

问题描述

给出n个数，找出这n个数的最大值，最小值，和。

输入格式

第一行为整数n，表示数的个数。

第二行有n个数，为给定的n个数，每个数的绝对值都小于10000。

输出格式

输出三行，每行一个整数。第一行表示这些数中的最大值，第二行表示这些数中的最小值，第三行表示这些数的和。

样例输入

5  
1 3 -2 4 5

样例输出

5  
-2  
11

数据规模与约定

1 <= n <= 10000。

#include<algorithm>

#include<cstring>

using namespace std;

int main(){

int n;

while(cin>>n){

int a[10005];

int sum=0;

for(int i=0;i<n;i++){

scanf("%d",&a[i]);

sum+=a[i];

}

sort(a,a+n);

cout<<a[n-1]<<endl<<a[0]<<endl<<sum<<endl;

}

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

#include <stdlib.h>

#define MAXSIZE 10000

int main()

{

int i;

int n;

int value;

int max=-MAXSIZE;

int min=MAXSIZE;

int sum=0;

scanf("%d",&n);

for(i=0;i<n;i++)

{

scanf("%d",&value);

if(value>max) max=value;

if(value<min) min=value;

sum+=value;

}

printf("%d\n",max);

printf("%d\n",min);

printf("%d\n",sum);

return 0;

}

**基础练习 字母图形**

问题描述

利用字母可以组成一些美丽的图形，下面给出了一个例子：

ABCDEFG

BABCDEF

CBABCDE

DCBABCD

EDCBABC

这是一个5行7列的图形，请找出这个图形的规律，并输出一个n行m列的图形。

输入格式

输入一行，包含两个整数n和m，分别表示你要输出的图形的行数的列数。

输出格式

输出n行，每个m个字符，为你的图形。

样例输入

5 7

样例输出

ABCDEFG  
BABCDEF  
CBABCDE  
DCBABCD  
EDCBABC

数据规模与约定

1 <= n, m <= 26。

本题的C++参考代码如下：

#include <iostream>

#include <cmath>

using namespace std;

int main()

{

int n, m;

cin >> n >> m;

for (int i = 0; i < n; ++i)

{

for (int j = 0; j < m; ++j)

cout << char('A'+abs(i-j));

cout << endl;

}

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

#include <stdlib.h>

#define MAXSIZE 26

//输出

void Print(char matrix[MAXSIZE][MAXSIZE],int n,int m)

{

int i,j;

for(i=0;i<n;i++)

{

for(j=0;j<m;j++)

{

printf("%c",matrix[i][j]);

}

printf("\n");

}

}

//实现字母图形

void Drawing(int n,int m)

{

int i,j;

int point=0;

char str;

char matrix[MAXSIZE][MAXSIZE];

for(i=0;i<n;i++)

{

str='A';

for(j=i;j<m;j++)

{

matrix[i][j]=str++;

}

str='A';

for(j=i-1;j>=0;j--)

{

matrix[i][j]=++str;

}

}

Print(matrix,n,m);

}

int main()

{

int n,m;

scanf("%d%d",&n,&m);

Drawing(n,m);

return 0;

}

**基础练习 01字串**

问题描述

对于长度为5位的一个01串，每一位都可能是0或1，一共有32种可能。它们的前几个是：

00000

00001

00010

00011

00100

请按从小到大的顺序输出这32种01串。

输入格式

本题的C++参考代码如下

#include <iostream>

using namespace std;

int main()

{

for (int i = 0; i <= 1; ++i)

for (int j = 0; j <= 1; ++j)

for (int k = 0; k <= 1; ++k)

for (int l = 0; l <= 1; ++l)

for (int m = 0; m <= 1; ++m)

cout << i << j << k << l << m << endl;

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

int main()

{

int i, j, k, l, m;

for (i = 0; i <= 1; ++i)

for (j = 0; j <= 1; ++j)

for (k = 0; k <= 1; ++k)

for (l = 0; l <= 1; ++l)

for (m = 0; m <= 1; ++m)

printf("%d%d%d%d%d\n", i, j, k, l, m);

return 0;

}

问题描述

给定一个年份，判断这一年是不是闰年。

当以下情况之一满足时，这一年是闰年：

1. 年份是4的倍数而不是100的倍数；

2. 年份是400的倍数。

其他的年份都不是闰年。

输入格式

输入包含一个整数y，表示当前的年份。

输出格式

输出一行，如果给定的年份是闰年，则输出yes，否则输出no。

说明：当试题指定你输出一个字符串作为结果（比如本题的yes或者no，你需要严格按照试题中给定的大小写，写错大小写将不得分。

样例输入

2013

样例输出

no

样例输入

2016

样例输出

yes

数据规模与约定

本题的C++参考代码如下：

1990 <= y <= 2050。

#include <iostream>

using namespace std;

int main()

{

int y;

cin >> y;

if (y%4==0 && y%100!=0 || y%400==0)

cout << "yes" << endl;

else

cout << "no" << endl;

return 0;

}

本题的C参考代码如下

#include <stdio.h>

int main()

{

int y;

scanf("%d", &y);

if (y%4==0 && y%100!=0 || y%400==0)

printf("yes");

else

printf("no");

return 0;

}

**基础练习 阶乘计算**

问题描述

　　输入一个正整数*n*，输出*n*!的值。  
　　其中*n*!=1\*2\*3\*…\**n*。

算法描述

*n*!可能很大，而计算机能表示的整数范围有限，需要使用高精度计算的方法。使用一个数组*A*来表示一个大整数*a*，*A*[0]表示*a*的个位，*A*[1]表示*a*的十位，依次类推。  
　　将*a*乘以一个整数*k*变为将数组*A*的每一个元素都乘以*k*，请注意处理相应的进位。  
　　首先将*a*设为1，然后乘2，乘3，当乘到*n*时，即得到了*n*!的值。

输入格式

　　输入包含一个正整数*n*，*n*<=1000。

输出格式

　　输出*n*!的准确值。

样例输入

10

样例输出

3628800

本题的C++参考代码如下：

#include <stdio.h>

#include <string.h>

#define MAX 10000

#define mod 10000

#define baselen 4

#define in(a) scanf("%d",&a)

#define out1(a) printf("%d",a)

#define out2(a) printf("%04d",a)

typedef int type;

struct bint{

type dig[MAX], len;

bint(){len = 0, dig[0] = 0;}

};

void by(bint a, type b, bint& c){

type i, carry;

for( i = carry = 0; i <= a.len || carry; i++){

if( i <= a.len ) carry += b\*a.dig[i];

c.dig[i] = carry%mod;

carry /= mod;

}

i--;

while( i && !c.dig[i] )i--;

c.len = i;

}

bool input(bint& a){

type i, j, w, k, p;

char data[MAX\*baselen+1];

if(scanf("%s",data)==EOF)return false;

w = strlen(data) - 1, a.len = 0;

for(p=0;p<=w&&data[p]=='0';p++);

while(1){

i = j = 0, k = 1;

while(i<baselen&&w>=p){

j = j+ (data[w--] - '0')\*k;

k \*= 10, i++;

}

a.dig[a.len++] = j;

if(w<p)break;

}

a.len--;

return true;

}

void output(bint& a){

type i;

i = a.len - 1;

out1(a.dig[a.len]);

while(i>=0)out2(a.dig[i--]);

}

void give(type a, bint& b){

b.dig[0] = a%mod;

a /= mod;

if(a>0)b.dig[1] = a, b.len = 1;

else b.len = 0;

}

int main()

{

bint a;int b,i;scanf("%d",&b);give(1,a);

for(i=2;i<=b;i++)by(a,i,a);

output(a);printf("\n");

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

#define N 10000

int main()

{

int a[N]={1};

int k=0,l=1,n;

int i,j;

scanf("%d",&n);

for(i=1;i<=n;i++)

{

for(j=0;j<l;j++)

{

a[j]=a[j]\*i+k;

k=a[j]/10000;

a[j]=a[j]%10000;

}

if(k)

{

a[j]=k;

l++;

k=0;

}

}

printf("%d",a[l-1]);

for(i=l-2;i>=0;i--)

printf("%04d",a[i]);

printf("\n");

return 0;

}

**基础练习 高精度加法**

问题描述

　　输入两个整数*a*和*b*，输出这两个整数的和。*a*和*b*都不超过100位。

算法描述

　　由于*a*和*b*都比较大，所以不能直接使用语言中的标准数据类型来存储。对于这种问题，一般使用数组来处理。  
　　定义一个数组*A*，*A*[0]用于存储*a*的个位，*A*[1]用于存储*a*的十位，依此类推。同样可以用一个数组*B*来存储*b*。  
　　计算*c* = *a* + *b*的时候，首先将*A*[0]与*B*[0]相加，如果有进位产生，则把进位（即和的十位数）存入*r*，把和的个位数存入*C*[0]，即*C*[0]等于(*A*[0]+*B*[0])%10。然后计算*A*[1]与*B*[1]相加，这时还应将低位进上来的值*r*也加起来，即*C*[1]应该是*A*[1]、*B*[1]和*r*三个数的和．如果又有进位产生，则仍可将新的进位存入到*r*中，和的个位存到*C*[1]中。依此类推，即可求出*C*的所有位。  
　　最后将*C*输出即可。

输入格式

　　输入包括两行，第一行为一个非负整数*a*，第二行为一个非负整数*b*。两个整数都不超过100位，两数的最高位都不是0。

输出格式

　　输出一行，表示*a*+ *b*的值。

样例输入

20100122201001221234567890  
2010012220100122

样例输出

20100122203011233454668012

本题的C++参考代码如下：#include<cstdio>

#include<iostream>

#include<cstring>

using namespace std;

int a[401],alen,b[401],blen,c[400],clen;

char st[400];

int main()

{

int i,j,n,len;

scanf("%s",st);

alen= strlen(st);

for(i=1;i<=alen;i++)

a[i]= st[alen-i]-48;

scanf("%s",st);

blen= strlen(st);

for(i=1;i<=blen;i++)

b[i]= st[blen-i]-48;

if(alen>blen) clen=alen;

else clen=blen;

for(i=1;i<=clen;i++)

c[i]=a[i]+b[i];

for(i=1;i<=clen;i++)

{

if(c[i]>=10)

{

c[i+1]=c[i+1]+ c[i]/10;

c[i]=c[i]%10;

}

}

if(c[clen+1]>0)clen++;

for(i=clen;i>=1;i--)

{

printf("%d",c[i]);

}

printf("\n");

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

void f(char a[],char b[])

{

int w=200,i,j,la,lb;

la=strlen(a);lb=strlen(b);

char c[200];

for(i=la;i<w;i++)a[i]=48;

for(i=lb;i<w;i++)b[i]=48;

a[w-1]=0;b[w-1]=0;

for(i=0;i<la;i++)c[i]=a[i];

for(i=0;i<w-1-la;i++)a[i]=48;j=0;

for(i=w-1-la;i<w-1;i++){a[i]=c[j];j++;}

for(i=0;i<lb;i++)c[i]=b[i];

for(i=0;i<w-1-lb;i++)b[i]=48;j=0;

for(i=w-1-lb;i<w-1;i++){b[i]=c[j];j++;}

for(i=w;i>=0;i--)

{

j=a[i]+b[i]-96;

if(j>9)a[i-1]=a[i-1]+j/10;

c[i]=j%10+48;

}

c[w-1]=0;

for(i=0;i<w;i++)if(c[i]!='0')break;

for(;i<w-1;i++)printf("%c",c[i]);printf("\n");

}

int main()

{

char a[200],b[200];gets(a);gets(b);

f(a,b);

return 0;

}

**基础练习 Huffuman树**

问题描述

　　Huffman树在编码中有着广泛的应用。在这里，我们只关心Huffman树的构造过程。  
　　给出一列数{*pi*}={*p*0, *p*1, …, *pn*-1}，用这列数构造Huffman树的过程如下：  
　　1. 找到{*pi*}中最小的两个数，设为*pa*和*pb*，将*pa*和*pb*从{*pi*}中删除掉，然后将它们的和加入到{*pi*}中。这个过程的费用记为*pa* +*pb*。  
　　2. 重复步骤1，直到{*pi*}中只剩下一个数。  
　　在上面的操作过程中，把所有的费用相加，就得到了构造Huffman树的总费用。  
　　本题任务：对于给定的一个数列，现在请你求出用该数列构造Huffman树的总费用。  
  
　　例如，对于数列{*pi*}={5, 3, 8, 2, 9}，Huffman树的构造过程如下：  
　　1. 找到{5, 3, 8, 2, 9}中最小的两个数，分别是2和3，从{*pi*}中删除它们并将和5加入，得到{5, 8, 9, 5}，费用为5。  
　　2. 找到{5, 8, 9, 5}中最小的两个数，分别是5和5，从{*pi*}中删除它们并将和10加入，得到{8, 9, 10}，费用为10。  
　　3. 找到{8, 9, 10}中最小的两个数，分别是8和9，从{*pi*}中删除它们并将和17加入，得到{10, 17}，费用为17。  
　　4. 找到{10, 17}中最小的两个数，分别是10和17，从{*pi*}中删除它们并将和27加入，得到{27}，费用为27。  
　　5. 现在，数列中只剩下一个数27，构造过程结束，总费用为5+10+17+27=59。

输入格式

　　输入的第一行包含一个正整数*n*（*n*<=100）。  
　　接下来是*n*个正整数，表示*p*0, *p*1, …, *pn*-1，每个数不超过1000。

输出格式

　　输出用这些数构造Huffman树的总费用。

样例输入

5  
5 3 8 2 9

样例输出

59

本题的C++参考代码如下：

#include<iostream>

#include<queue>

using namespace std;

priority\_queue<int, vector<int>, greater<int> > pq; //构造从小到大的优先队列

int main() {

int n;

cin >> n;

while (!pq.empty())

pq.pop();

int x, s;

for (int i = 0; i < n; i++) {

cin >> x;

pq.push(x);

}

int sum = 0;

while (pq.size() > 1) {

s = pq.top();

pq.pop();

s += pq.top();

pq.pop();

sum += s;

pq.push(s);

}

cout << sum << endl;

}

本题的C参考代码如下：

#include <stdio.h>

typedef struct

{

int a[100];

int len;

}huf;

int sum=0;

int del(huf\* in,int t)

{

int i,j;

for(i=0;i<in->len && in->a[i]!=t;i++);

for(;i<in->len-1;i++)

in->a[i]=in->a[i+1];

in->len--;

return 1;

}

int add(huf\* in,int t)

{

in->a[in->len]=t;

in->len++;

}

int find\_two\_mins(huf\* in)

{

int i,j,t;

int mina,minb;

for(i=0;i<in->len-1;i++)

for(j=i+1;j<in->len;j++)

if(in->a[i]>in->a[j])

{

t=in->a[i];

in->a[i]=in->a[j];

in->a[j]=t;

}

mina=in->a[0];

minb=in->a[1];

del(in,mina);

del(in,minb);

add(in,mina+minb);

return mina+minb;

}

int main()

{

huf in;

int i,j,n;

scanf("%d",&n);

in.len=n;

for(i=0;i<n;i++)

scanf("%d",&in.a[i]);

while(1)

{

if(in.len==2)

{

sum=sum+in.a[0]+in.a[1];

break;

}

sum+=find\_two\_mins(&in);

}

printf("%d",sum);

return 0;

}

**基础练习 2n皇后问题**

问题描述

　　给定一个n\*n的棋盘，棋盘中有一些位置不能放皇后。现在要向棋盘中放入n个黑皇后和n个白皇后，使任意的两个黑皇后都不在同一行、同一列或同一条对角线上，任意的两个白皇后都不在同一行、同一列或同一条对角线上。问总共有多少种放法？n小于等于8。

输入格式

　　输入的第一行为一个整数n，表示棋盘的大小。  
　　接下来n行，每行n个0或1的整数，如果一个整数为1，表示对应的位置可以放皇后，如果一个整数为0，表示对应的位置不可以放皇后。

输出格式

　　输出一个整数，表示总共有多少种放法。

样例输入

4  
1 1 1 1  
1 1 1 1  
1 1 1 1  
1 1 1 1

样例输出

2

样例输入

4  
1 0 1 1  
1 1 1 1  
1 1 1 1  
1 1 1 1

样例输出

0

#include<cstdio>

using namespace std;

int n;

int sum;

bool g[9][9];

bool wh[9];

bool wd[17];

bool wu[17];

bool bh[9];

bool bd[17];

bool bu[17];

void white(int h){

if(h==n){

sum++;

}else{

for(int i=0;i<n;i++){

if(!g[h][i])continue;

if(wh[i])continue;

if(wd[i+h])continue;

if(wu[(i-h)+n])continue;

wh[i]=wd[i+h]=wu[(i-h)+n]=1;

white(h+1);

wh[i]=wd[i+h]=wu[(i-h)+n]=0;

}

}

}

void black(int h){

if(h==n){

white(0);

}else{

for(int i=0;i<n;i++){

if(!g[h][i])continue;

if(bh[i])continue;

if(bd[i+h])continue;

if(bu[(i-h)+n])continue;

g[h][i]=0;

bh[i]=bd[i+h]=bu[(i-h)+n]=1;

black(h+1);

g[h][i]=1;

bh[i]=bd[i+h]=bu[(i-h)+n]=0;

}

}

}

int main(){

int i;

int x;

sum=0;

scanf("%d",&n);

for(i=0;i<n;i++){

wh[i]=bh[i]=0;

wd[i]=bd[i]=0;

wu[i]=bu[i]=0;

for(int j=0;j<n;j++){

scanf("%d",&x);

g[i][j]=(bool)x;

}

}

for(;i<2\*n;i++){

wd[i]=bd[i]=0;

wu[i]=bu[i]=0;

}

black(0);

printf("%d\n",sum);

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

int a[9][9],n,sum;

int row1[9],diagl1[17],diagr1[17],row2[9],diagl2[17],diagr2[17];

void dfs2(int i)

{

int j;

if(i==n+1)

{

sum++;

return;

}

for(j=1;j<=n;j++)

{

if(a[i][j]==1&&row2[j]!=0&&diagl2[n-i+j]!=0&&diagr2[i+j]!=0)

{

a[i][j]=0;

row2[j]=0;

diagl2[n-i+j]=0;

diagr2[i+j]=0;

dfs2(i+1);

row2[j]=1;

diagl2[n-i+j]=1;

diagr2[i+j]=1;

a[i][j]=1;

}

}

}

void dfs1(int i)

{

int j,h,r;

if(i==n+1)

{

for(h=1,r=1;r<=n;r++)

{

if(a[1][r]==1)

{

a[1][r]=0;

row2[r]=0;

diagl2[n-h+r]=0;

diagr2[h+r]=0;

dfs2(h+1);

row2[r]=1;

diagl2[n-h+r]=1;

diagr2[h+r]=1;

a[1][r]=1;

}

}

}

for(j=1;j<=n;j++)

{

if(a[i][j]==1&&row1[j]!=0&&diagl1[n-i+j]!=0&&diagr1[i+j]!=0)

{

a[i][j]=0;

row1[j]=0;

diagl1[n-i+j]=0;

diagr1[i+j]=0;

dfs1(i+1);

row1[j]=1;

diagl1[n-i+j]=1;

diagr1[i+j]=1;

a[i][j]=1;

}

}

}

int main()

{

int i,j;

while(scanf("%d",&n)!=EOF)

{

sum=0;

for(i=1;i<=n;i++)

for(j=1;j<=n;j++)

scanf("%d",&a[i][j]);

for(i=1;i<=n;i++)

row1[i]=row2[i]=1;

for(i=1;i<=2\*n;i++)

diagl1[i]=diagr1[i]=diagl2[i]=diagr2[i]=1;

for(i=1,j=1;j<=n;j++)

{

if(a[1][j]==1)

{

a[1][j]=0;

row1[j]=0;

diagl1[n-i+j]=0;

diagr1[i+j]=0;

dfs1(i+1);

row1[j]=1;

diagl1[n-i+j]=1;

diagr1[i+j]=1;

a[1][j]=1;

}

}

printf("%d\n",sum);

}

return 0;

}

**基础练习 报时助手**

问题描述

　　给定当前的时间，请用英文的读法将它读出来。  
　　时间用时h和分m表示，在英文的读法中，读一个时间的方法是：  
　　如果m为0，则将时读出来，然后加上“o'clock”，如3:00读作“three o'clock”。  
　　如果m不为0，则将时读出来，然后将分读出来，如5:30读作“five thirty”。  
　　时和分的读法使用的是英文数字的读法，其中0~20读作：  
　　0:zero, 1: one, 2:two, 3:three, 4:four, 5:five, 6:six, 7:seven, 8:eight, 9:nine, 10:ten, 11:eleven, 12:twelve, 13:thirteen, 14:fourteen, 15:fifteen, 16:sixteen, 17:seventeen, 18:eighteen, 19:nineteen, 20:twenty。  
　　30读作thirty，40读作forty，50读作fifty。  
　　对于大于20小于60的数字，首先读整十的数，然后再加上个位数。如31首先读30再加1的读法，读作“thirty one”。  
　　按上面的规则21:54读作“twenty one fifty four”，9:07读作“nine seven”，0:15读作“zero fifteen”。

输入格式

　　输入包含两个非负整数h和m，表示时间的时和分。非零的数字前没有前导0。h小于24，m小于60。

输出格式

　　输出时间时刻的英文。

样例输入

0 15

样例输出

zero fifteen

本题的C++参考代码如下：

#include <iostream>

#include <string>

#include <map>

using namespace std;

int main(int argc, char\*\* argv)

{

map<int,string> maptime;

maptime[0]="zero";

maptime[1]="one";

maptime[2]="two";

maptime[3]="three";

maptime[4]="four";

maptime[5]="five";

maptime[6]="six";

maptime[7]="seven";

maptime[8]="eight";

maptime[9]="nine";

maptime[10]="ten";

maptime[11]="eleven";

maptime[12]="twelve";

maptime[13]="thirteen";

maptime[14]="fourteen";

maptime[15]="fifteen";

maptime[16]="sixteen";

maptime[17]="seventeen";

maptime[18]="eighteen";

maptime[19]="nineteen";

maptime[20]="twenty";

maptime[30]="thirty";

maptime[40]="forty";

maptime[50]="fifty";

int h,m;

cin>>h>>m;

if(m==0)

{

if(h<=20)

{

cout<<maptime[h]<<" o'clock";

}

else

{

cout<<maptime[20]<<" "<<maptime[h-20]<<" o'clock";

}

}

else

{

if(h<=20)

{

cout<<maptime[h]<<" ";

}

else

{

cout<<maptime[20]<<" "<<maptime[h-20]<<" ";

}

if(m<=20)

{

cout<<maptime[m]<<" ";

}

else

{

int k=m%10;

cout<<maptime[m-k]<<" "<<maptime[k]<<" ";

}

}

return 0;

}  
本题的C参考代码如下：

#include<stdio.h>

#include<string.h>

int main()

{

int h,m;

char g[50][50]={"zero","one","two","three","four","five","six","seven","eight","nine","ten","eleven","twelve","thirteen","fourteen","fifteen","sixteen","seventeen","eighteen","nineteen","twenty"};

char s[10][10]={"twenty","thirty","forty","fifty"};

scanf("%d%d",&h,&m);

if(m==0)if(h<21)printf("%s o'clock",g[h]);else {printf("%s %s o'clock",s[h/10-2],g[h%10]);}

else {

if(h<21)printf("%s ",g[h]);else {printf("%s %s ",s[h/10-2],g[h%10]);}

if(m<21)printf("%s ",g[m]);else {printf("%s %s ",s[m/10-2],g[m%10]);}

}

printf("\n");

return 0;

}

**基础练习 回形取数**

问题描述

　　回形取数就是沿矩阵的边取数，若当前方向上无数可取或已经取过，则左转90度。一开始位于矩阵左上角，方向向下。

输入格式

　　输入第一行是两个不超过200的正整数m, n，表示矩阵的行和列。接下来m行每行n个整数，表示这个矩阵。

输出格式

　　输出只有一行，共mn个数，为输入矩阵回形取数得到的结果。数之间用一个空格分隔，行末不要有多余的空格。

样例输入

3 3  
1 2 3  
4 5 6  
7 8 9

样例输出

1 4 7 8 9 6 3 2 5

样例输入

3 2  
1 2  
3 4  
5 6

样例输出

1 3 5 6 4 2

本题的C++参考代码如下：

#include<cstdio>

#include<cstring>

#include<cstdlib>

#include<iostream>

using namespace std;

int m, n;

int a[210][210]={0};

int ans[44100];

int t=0;

void sr()

{

scanf("%d %d",&m,&n);

int i,j;

for(i=0;i<m;i++)for(j=0;j<n;j++)scanf("%d",&a[i][j]);

}

void xx(int x1,int y1,int x,int y)

{

int i,j;

if(x==0||y==0)return;

if(x==1)

{

for(i=0;i<y;i++)

{

ans[t]=a[x1][y1];

y1++;

t++;

}

return;

}

if(y==1)

{

for(i=0;i<x;i++)

{

ans[t]=a[x1][y1];

x1++;

t++;

}

return;

}

int v=x1,h=y1;

ans[t]=a[x1][y1];

t++;

for(i=0;i<x-1;i++){v++;ans[t]=a[v][h];t++;}

for(i=0;i<y-1;i++){h++;ans[t]=a[v][h];t++;}

for(i=0;i<x-1;i++){v--;ans[t]=a[v][h];t++;}

for(i=0;i<y-2;i++){h--;ans[t]=a[v][h];t++;}

xx(v+1,h,x-2,y-2);

}

void sc()

{

int c=n\*m;

if(c==0)return;

printf("%d",ans[0]);

for(int i=1;i<c;i++)printf(" %d",ans[i]);

}

int main()

{

sr();

int x=m,y=n;

xx(0,0,x,y);

sc();

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

#include <string.h>

#define MAX\_N 200

int m,n;

int a[MAX\_N][MAX\_N],b[MAX\_N][MAX\_N];

int s=0;

void solve(int i,int j)

{

if(i<m && i>=0 && j<n && j>=0 && b[i][j] == 0)

{

printf("%d ",a[i][j]);

b[i][j] = 1;

}

else

{

s++;

return ;

}

if(s%4 == 0)

solve(i+1,j);

if(s%4 == 1)

solve(i,j+1);

if(s%4 == 2)

solve(i-1,j);

if(s%4 == 3)

solve(i,j-1);

if(s%4 == 0)

solve(i+1,j);

if(s%4 == 1)

solve(i,j+1);

if(s%4 == 2)

solve(i-1,j);

if(s%4 == 3)

solve(i,j-1);

return ;

}

int main()

{

memset(b,0,sizeof(b));

scanf("%d%d",&m,&n);

int i,j;

for(i=0; i<m; i++)

for(j=0; j<n; j++)

scanf("%d",&a[i][j]);

solve(0,0);

puts("");

return 0;

}

**基础练习 龟兔赛跑预测**

问题描述

　　话说这个世界上有各种各样的兔子和乌龟，但是研究发现，所有的兔子和乌龟都有一个共同的特点——喜欢赛跑。于是世界上各个角落都不断在发生着乌龟和兔子的比赛，小华对此很感兴趣，于是决定研究不同兔子和乌龟的赛跑。他发现，兔子虽然跑比乌龟快，但它们有众所周知的毛病——骄傲且懒惰，于是在与乌龟的比赛中，一旦任一秒结束后兔子发现自己领先t米或以上，它们就会停下来休息s秒。对于不同的兔子，t，s的数值是不同的，但是所有的乌龟却是一致——它们不到终点决不停止。  
　　然而有些比赛相当漫长，全程观看会耗费大量时间，而小华发现只要在每场比赛开始后记录下兔子和乌龟的数据——兔子的速度v1（表示每秒兔子能跑v1米），乌龟的速度v2，以及兔子对应的t，s值，以及赛道的长度l——就能预测出比赛的结果。但是小华很懒，不想通过手工计算推测出比赛的结果，于是他找到了你——清华大学计算机系的高才生——请求帮助，请你写一个程序，对于输入的一场比赛的数据v1，v2，t，s，l，预测该场比赛的结果。

输入格式

　　输入只有一行，包含用空格隔开的五个正整数v1，v2，t，s，l，其中(v1,v2<=100;t<=300;s<=10;l<=10000且为v1,v2的公倍数)

输出格式

　　输出包含两行，第一行输出比赛结果——一个大写字母“T”或“R”或“D”，分别表示乌龟获胜，兔子获胜，或者两者同时到达终点。  
　　第二行输出一个正整数，表示获胜者（或者双方同时）到达终点所耗费的时间（秒数）。

样例输入

10 5 5 2 20

样例输出

D  
4

样例输入

10 5 5 1 20

样例输出

R  
3

样例输入

10 5 5 3 20

样例输出

T  
4

本题的C++参考代码如下：

#include<cstdio>

#include<iostream>

using namespace std;

int main()

{

int v1,v2,t,s,L,L1=0,L2=0,ans=0,i,j;

bool bk=true;

scanf("%d%d%d%d%d",&v1,&v2,&t,&s,&L);

while(1)

{

if(L1-L2>=t&&L1<L&&L2<L)

{

for(i=1;i<=s;i++)

{

if(L1<L&&L2<L)

{

L2+=v2;

ans++;

}

}

}

else

{

L2+=v2;

L1+=v1;

ans++;

}

if(L1>=L) break;

if(L2>=L) break;

}

if(L1>=L&&L2>=L){printf("D\n%d\n",ans);return 0;}

if(L1>=L) {printf("R\n%d\n",ans); return 0;}

if(L2>=L) {printf("T\n%d\n",ans); return 0;}

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

int main()

{

int v1,v2,t,s,l,s1=0,s2=0,i=0;

scanf("%d %d %d %d %d",&v1,&v2,&t,&s,&l);

while(s1<l&&s2<l)

{

s1+=v1;

s2+=v2;

i++;

if(s1==l||s2==l)break;

if(s1-s2>=t)s1-=v1\*s;

}

if(s1>s2) printf("R\n");

else if(s2>s1) printf("T\n");

else printf("D\n");

printf("%d",i);

return 0;

}

**基础练习 芯片测试**

问题描述

　　有n（2≤n≤20）块芯片，有好有坏，已知好芯片比坏芯片多。  
　　每个芯片都能用来测试其他芯片。用好芯片测试其他芯片时，能正确给出被测试芯片是好还是坏。而用坏芯片测试其他芯片时，会随机给出好或是坏的测试结果（即此结果与被测试芯片实际的好坏无关）。  
　　给出所有芯片的测试结果，问哪些芯片是好芯片。

输入格式

　　输入数据第一行为一个整数n，表示芯片个数。  
　　第二行到第n+1行为n\*n的一张表，每行n个数据。表中的每个数据为0或1，在这n行中的第i行第j列（1≤i, j≤n）的数据表示用第i块芯片测试第j块芯片时得到的测试结果，1表示好，0表示坏，i=j时一律为1（并不表示该芯片对本身的测试结果。芯片不能对本身进行测试）。

输出格式

　　按从小到大的顺序输出所有好芯片的编号

样例输入

3  
1 0 1  
0 1 0  
1 0 1

样例输出

1 3

本题的C++参考代码如下：

#include<iostream>

#include<cstdio>

#include<cstring>

using namespace std;

bool a[25][25];

bool v[25];

int n;

bool dfs(int k)

{

if (k==n)

{

int sum=0;

for (int i=1;i<=n;i++)

if (v[i])

sum++;

if (sum>n-sum)

for (int i=1;i<=n;i++)

if (v[i])

printf("%d ",i);

return true;

}

if (v[k]==true)

{

int len=0,s[25];

for (int i=1;i<=n;i++)

if (!a[k][i] && v[i])

{

s[++len]=i;

v[i]=false;

}

if (dfs(k+1)) return true;

for (int i=1;i<=len;i++)

v[s[i]]=true;

}

if (dfs(k+1)) return true;

}

int main()

{

scanf("%d",&n);

memset(v,true,sizeof(v));

for (int i=1;i<=n;i++)

for (int j=1;j<=n;j++)

{

int c;

scanf("%d",&c);

if (c) a[i][j]=1; else a[i][j]=0;

}

dfs(1);

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

#include<string.h>

int main()

{

int a[50][50];int i,j,n,s;scanf("%d",&n);

for(j=0;j<n;j++)for(i=0;i<n;i++)scanf("%d",&a[i][j]);

for(j=0;j<n;j++){s=0;for(i=0;i<n;i++)s=s+a[j][i];if(s>n/2.0)printf("%d ",j+1);}

return 0;

}

**基础练习 FJ的字符串**

问题描述

　　FJ在沙盘上写了这样一些字符串：  
　　A1 = “A”  
　　A2 = “ABA”  
　　A3 = “ABACABA”  
　　A4 = “ABACABADABACABA”  
　　… …  
　　你能找出其中的规律并写所有的数列AN吗？

输入格式

　　仅有一个数：N ≤ 26。

输出格式

　　请输出相应的字符串AN，以一个换行符结束。输出中不得含有多余的空格或换行、回车符。

样例输入

3

样例输出

ABACABA

本题的C++参考代码如下：

#include<iostream>

#include<cstdio>

using namespace std;

void dfs(int k,int p)

{

if (k==1)

{

printf("%c",p+'A');

return;

}

dfs(k/2,p-1);dfs(1,p);dfs(k/2,p-1);

}

int main()

{

int n;

scanf("%d",&n);

int sum=1;

n--;

for (int i=1;i<=n;i++) sum=sum\*2+1;

dfs(sum,n);

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

void f(int a)

{

if(a==0)printf("%c",'A');

else

{

f(a-1);

printf("%c",a+'A');

f(a-1);

}

}

int main()

{

int a;scanf("%d",&a);f(a-1);printf("\n");

return 0;

}

**基础练习 Sine之舞**

问题描述

　　最近FJ为他的奶牛们开设了数学分析课，FJ知道若要学好这门课，必须有一个好的三角函数基本功。所以他准备和奶牛们做一个“Sine之舞”的游戏，寓教于乐，提高奶牛们的计算能力。  
　　不妨设  
　　An=sin(1–sin(2+sin(3–sin(4+...sin(n))...)  
　　Sn=(...(A1+n)A2+n-1)A3+...+2)An+1  
　　FJ想让奶牛们计算Sn的值，请你帮助FJ打印出Sn的完整表达式，以方便奶牛们做题。

输入格式

　　仅有一个数：N<201。

输出格式

　　请输出相应的表达式Sn，以一个换行符结束。输出中不得含有多余的空格或换行、回车符。

样例输入

3

样例输出

((sin(1)+3)sin(1–sin(2))+2)sin(1–sin(2+sin(3)))+1

本题的C++参考代码如下：

#include<stdio.h>

void An\_Output(int n, int t)

{

if(n == t)

{

printf("sin(%d)", t);

return ;

}

char c;

c = t % 2 == 1 ? '+' : '-';

printf("sin(%d%c", t, c);

An\_Output(n, ++t);

printf(")");

}

void Sn\_Output(int n, int t)

{

//　Sn=(...(A1+n)A2+n-1)A3+...+2)An+1

if(n == t)

{

return ;

}

printf("(");

Sn\_Output(n, t+1);

if(t != n - 1)

{

printf(")");

}

An\_Output(n - t, 1);

printf("+%d", t+1);

}

int main()

{

int n;

scanf("%d", &n);

Sn\_Output(n, 1);

if(n!=1)

printf(")");

An\_Output(n, 1);

printf("+1\n");

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

void printA(int n, int k)

{

if (n == k)

printf("sin(%d)", n);

else

{

printf("sin(%d", n);

printf(n % 2 == 0 ? "+" : "-");

printA(n + 1, k);

printf(")");

}

}

void printS(int n, int k)

{

if (n == 1)

{

printA(1, n);

printf("+%d", k - n);

}

else

{

printf("(");

printS(n - 1, k);

printf(")");

printA(1, n);

printf("+%d", k - n);

}

}

int main()

{

int N;

scanf("%d", &N);

printS(N, N + 1);

printf("\n");

return 0;

}

**基础练习 数的读法**

问题描述

　　Tom教授正在给研究生讲授一门关于基因的课程，有一件事情让他颇为头疼：一条染色体上有成千上万个碱基对，它们从0开始编号，到几百万，几千万，甚至上亿。  
　　比如说，在对学生讲解第1234567009号位置上的碱基时，光看着数字是很难准确的念出来的。  
　　所以，他迫切地需要一个系统，然后当他输入12 3456 7009时，会给出相应的念法：  
　　十二亿三千四百五十六万七千零九  
　　用汉语拼音表示为  
　　shi er yi san qian si bai wu shi liu wan qi qian ling jiu  
　　这样他只需要照着念就可以了。  
　　你的任务是帮他设计这样一个系统：给定一个阿拉伯数字串，你帮他按照中文读写的规范转为汉语拼音字串，相邻的两个音节用一个空格符格开。  
　　注意必须严格按照规范，比如说“10010”读作“yi wan ling yi shi”而不是“yi wan ling shi”，“100000”读作“shi wan”而不是“yi shi wan”，“2000”读作“er qian”而不是“liang qian”。

输入格式

　　有一个数字串，数值大小不超过2,000,000,000。

输出格式

　　是一个由小写英文字母，逗号和空格组成的字符串，表示该数的英文读法。

样例输入

1234567009

样例输出

shi er yi san qian si bai wu shi liu wan qi qian ling jiu

本题的C++参考代码如下：

#include<iostream>

#include<cstdio>

#include<cstring>

#include<cstdlib>

using namespace std;

char df[][10]={"ling","yi","er","san","si","wu","liu","qi","ba","jiu"};

char s[15];

int main()

{

scanf("%s",s);

int lens=strlen(s);

bool bk=false;

for (int i=0;i<lens;i++)

{

int p,lendf;

p=s[i]-'0';

if (p!=0)

{

bk=false;

lendf=strlen(df[p]);

if (s[i-1]-'0'==0)

printf("ling ");

if ((lens-i)%4==2 && p==1 /\*&& s[i-1]-'0'==0 && s[i-2]-'0'==0\*/ && i==0)

{

printf("shi ");

continue;

}

for (int j=0;j<lendf;j++)

printf("%c",df[p][j]);

printf(" ");

if ((lens-i)%4==2) printf("shi ");

if ((lens-i)%4==3) printf("bai ");

if ((lens-i)%4==0) printf("qian ");

}

if ((lens-i)%4==1)

{

if ((lens-i)/4==2)

{

bk=true;

printf("yi ");

}

if (bk==false && (lens-i)/4==1) printf("wan ");

}

}

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

#include<string.h>

int main()

{

char a[100];int i,j,k,l;

char b[20][10]={"ling","yi","er","san","si","wu","liu","qi","ba","jiu"};

char c[20][10]={"","","shi","bai","qian","wan","shi","bai","qian","yi","shi"};

gets(a);

l=strlen(a);

for(i=0;i<l;i++){

j=a[i]-48;

if(j==0){if(i<l-1)if(a[i+1]!=48)printf("%s ",b[j]);}

else

if((l-i==2||l-i==6||l-i==10)&&j==1)printf("%s ",c[l-i]);

else if(a[i-1]==48&&j==1)printf("%s ",c[l-i]);

else printf("%s %s ",b[j],c[l-i]);

}

printf("\n");

return 0;

}

**基础练习 完美的代价**

问题描述

　　回文串，是一种特殊的字符串，它从左往右读和从右往左读是一样的。小龙龙认为回文串才是完美的。现在给你一个串，它不一定是回文的，请你计算最少的交换次数使得该串变成一个完美的回文串。  
　　交换的定义是：交换两个相邻的字符  
　　例如mamad  
　　第一次交换 ad : mamda  
　　第二次交换 md : madma  
　　第三次交换 ma : madam (回文！完美！)

输入格式

　　第一行是一个整数N，表示接下来的字符串的长度(N <= 8000)  
　　第二行是一个字符串，长度为N.只包含小写字母

输出格式

　　如果可能，输出最少的交换次数。  
　　否则输出Impossible

样例输入

5  
mamad

样例输出

3

本题的C++参考代码如下：

#include<cstdio>

#include<iostream>

using namespace std;

int changes(char s[],char x,int n)

{

int i,change=0,j,k;

for(i=0;i<n/2;i++)

{

if(s[i]==x)

{

for(j=i;j<n-i-1;j++)

if(s[n-i-1]==s[j])

break;

change+=j-i;

for(k=j;k>i;k--)

s[k]=s[k-1];

s[i]=s[n-i-1];

}

else

{

for(j=n-i-1;j>=i;j--)

if(s[i]==s[j])

break;

change+=n-i-1-j;

for(k=j;k<n-i-1;k++) s[k]=s[k+1];

s[n-i-1]=s[i];

}

}

return change;

}

int main()

{

int n,i,k=0,b[26]={0},j;

char y,s[8001]={0};

scanf("%d\n",&n);

for(i=0;i<n;i++)

{

scanf("%c",&s[i]);

b[s[i]-'a']++;

}

char x;

for(j=0;j<26;j++)

{

if(b[j]%2!=0)

{

k++;

x=j+'a';

}

}

if(k>=2)

printf("Impossible\n");

else

{

printf("%d\n",changes(s,x,n));

return 0;

}

}

本题的C参考代码如下：

#include<stdio.h>

int changes(char s[],char x,int n);

char x='0';

int main(void)

{

int n,i,k=0,b[26]={0},j;

char y,s[8000]={0};

scanf("%d",&n);

getchar();

for(i=0;i<n;i++)

scanf("%c",&s[i]);

for(i=0;i<n;i++)

{

j=s[i]-'a';

b[j]++;

}

for(j=0;j<26;j++)

{

if(b[j]%2!=0)

{

k++;

x=j+'a';

}

}

if(k>=2)

printf("Impossible\n");

else

printf("%d\n",changes(s,x,n));

return 0;

}

int changes(char s[],char x,int n)

{

int i,change=0,j,k;

for(i=0;i<n/2;i++)

{

if(s[i]==x)

{

for(j=i;j<n-i-1;j++)

if(s[n-i-1]==s[j])

break;

change+=j-i;

for(k=j;k>i;k--)

s[k]=s[k-1];

s[i]=s[n-i-1];

}

else

{

for(j=n-i-1;j>=i;j--)

if(s[i]==s[j])

break;

change+=n-i-1-j;

for(k=j;k<n-i-1;k++)

s[k]=s[k+1];

s[n-i-1]=s[i];

}

}

return change;

}

**基础练习 矩形面积交**

问题描述

　　平面上有两个矩形，它们的边平行于直角坐标系的X轴或Y轴。对于每个矩形，我们给出它的一对相对顶点的坐标，请你编程算出两个矩形的交的面积。

输入格式

　　输入仅包含两行，每行描述一个矩形。  
　　在每行中，给出矩形的一对相对顶点的坐标，每个点的坐标都用两个绝对值不超过10^7的实数表示。

输出格式

　　输出仅包含一个实数，为交的面积，保留到小数后两位。

样例输入

1 1 3 3  
2 2 4 4

样例输出

1.00

本题的C++参考代码如下：

#include <iostream>

#include <algorithm>

#include <cmath>

#include <cstdio>

using namespace std;

int main()

{

double x1,x2,y1,y2;

double q1,q2,w1,w2;

while(cin>>x1>>y1>>x2>>y2>>q1>>w1>>q2>>w2)

{

double xx=max(min(x1,x2),min(q1,q2));

double yy=max(min(y1,y2),min(w1,w2));

double xxup=min(max(x1,x2),max(q1,q2));

double yyup=min(max(y1,y2),max(w1,w2));

if(xxup>xx)

printf("%.2f\n",fabs((xx)-(xxup))\*fabs((yy)-(yyup)));

else printf("0.00\n");

}

}

本题的C参考代码如下：

//天农计算机系许晓华老师出品

#include <stdio.h>

#define max(x,y) ((x)>(y)?(x):(y))

#define min(x,y) ((x)<(y)?(x):(y))

int main()

{

double x1,y1,x2,y2;//矩形1

double x3,y3,x4,y4;//矩形2

double m1,n1;//交集左上角坐标

double m2,n2;//交集右下角坐标

scanf("%lf%lf%lf%lf",&x1,&y1,&x2,&y2);

scanf("%lf%lf%lf%lf",&x3,&y3,&x4,&y4);

m1=max(min(x1,x2),min(x3,x4));

n1=max(min(y1,y2),min(y3,y4));

m2=min(max(x1,x2),max(x3,x4));

n2=min(max(y1,y2),max(y3,y4));

if(m2>m1&&n2>n1)

printf("%.2f\n",(m2-m1)\*(n2-n1));

else

printf("0.00\n");

return 0;

}

**基础练习 矩阵乘法**

问题描述

　　给定一个N阶矩阵A，输出A的M次幂（M是非负整数）  
　　例如：  
　　A =  
　　1 2  
　　3 4  
　　A的2次幂  
　　7 10  
　　15 22

输入格式

　　第一行是一个正整数N、M（1<=N<=30, 0<=M<=5），表示矩阵A的阶数和要求的幂数  
　　接下来N行，每行N个绝对值不超过10的非负整数，描述矩阵A的值

输出格式

　　输出共N行，每行N个整数，表示A的M次幂所对应的矩阵。相邻的数之间用一个空格隔开

样例输入

2 2  
1 2  
3 4

样例输出

7 10  
15 22

本题的C++参考代码如下：

#include<cstdio>

#include<iostream>

#include<cstring>

using namespace std;

int a[101][101];

int c[101][101];

int ans[101][101];

int main()

{

int i,j,k,l,m,n;

scanf("%d%d",&n,&m);

for(i=1;i<=n;i++)

for(j=1;j<=n;j++)

scanf("%d",&a[i][j]);

memset(ans,0,sizeof(ans));

for(i=1;i<=n;i++) ans[i][i]=1;

for(k=1;k<=m;k++)

{ memset(c,0,sizeof(c));

for(i=1;i<=n;i++)for(j=1;j<=n;j++)for(l=1;l<=n;l++)c[i][j]+=ans[i][l]\*a[l][j];

for(i=1;i<=n;i++)for(j=1;j<=n;j++)ans[i][j]=c[i][j];

}

for(i=1;i<=n;i++)

{

for(j=1;j<n;j++)printf("%d ",ans[i][j]);

printf("%d\n",ans[i][n]);

}

return 0;

}

本题的C参考代码如下：

//天农计算机系许晓华老师出品

#include <stdio.h>

#include <string.h>

#define N 100

int A[N][N],t[N][N],r[N][N];

int main()

{

int n,m,i,j,k;

scanf("%d%d",&n,&m);

for(i=0;i<n;i++)

for(j=0;j<n;j++)

scanf("%d",&A[i][j]);

for(i=0;i<n;i++)

r[i][i]=1;//单位矩阵，如同数的乘法中的1

while(m--)

{

memset(t,0,sizeof(t));

for(i=0;i<n;i++)

for(j=0;j<n;j++)

for(k=0;k<n;k++)

t[i][j]+=r[i][k]\*A[k][j];

for(i=0;i<n;i++)

for(j=0;j<n;j++)

r[i][j]=t[i][j];

}

for(i=0;i<n;i++)

{

for(j=0;j<n-1;j++)

printf("%d ",r[i][j]);

printf("%d\n",r[i][j]);

}

return 0;

}

**基础练习 矩阵乘法**

问题描述

　　给定一个N阶矩阵A，输出A的M次幂（M是非负整数）  
　　例如：  
　　A =  
　　1 2  
　　3 4  
　　A的2次幂  
　　7 10  
　　15 22

输入格式

　　第一行是一个正整数N、M（1<=N<=30, 0<=M<=5），表示矩阵A的阶数和要求的幂数  
　　接下来N行，每行N个绝对值不超过10的非负整数，描述矩阵A的值

输出格式

　　输出共N行，每行N个整数，表示A的M次幂所对应的矩阵。相邻的数之间用一个空格隔开

样例输入

2 2  
1 2  
3 4

样例输出

7 10  
15 22

本题的C++参考代码如下：

#include<cstdio>

#include<iostream>

#include<cstring>

using namespace std;

int a[101][101];

int c[101][101];

int ans[101][101];

int main()

{

int i,j,k,l,m,n;

scanf("%d%d",&n,&m);

for(i=1;i<=n;i++)

for(j=1;j<=n;j++)

scanf("%d",&a[i][j]);

memset(ans,0,sizeof(ans));

for(i=1;i<=n;i++) ans[i][i]=1;

for(k=1;k<=m;k++)

{ memset(c,0,sizeof(c));

for(i=1;i<=n;i++)for(j=1;j<=n;j++)for(l=1;l<=n;l++)c[i][j]+=ans[i][l]\*a[l][j];

for(i=1;i<=n;i++)for(j=1;j<=n;j++)ans[i][j]=c[i][j];

}

for(i=1;i<=n;i++)

{

for(j=1;j<n;j++)printf("%d ",ans[i][j]);

printf("%d\n",ans[i][n]);

}

return 0;

}

本题的C参考代码如下：

//天农计算机系许晓华老师出品

#include <stdio.h>

#include <string.h>

#define N 100

int A[N][N],t[N][N],r[N][N];

int main()

{

int n,m,i,j,k;

scanf("%d%d",&n,&m);

for(i=0;i<n;i++)

for(j=0;j<n;j++)

scanf("%d",&A[i][j]);

for(i=0;i<n;i++)

r[i][i]=1;//单位矩阵，如同数的乘法中的1

while(m--)

{

memset(t,0,sizeof(t));

for(i=0;i<n;i++)

for(j=0;j<n;j++)

for(k=0;k<n;k++)

t[i][j]+=r[i][k]\*A[k][j];

for(i=0;i<n;i++)

for(j=0;j<n;j++)

r[i][j]=t[i][j];

}

for(i=0;i<n;i++)

{

for(j=0;j<n-1;j++)

printf("%d ",r[i][j]);

printf("%d\n",r[i][j]);

}

return 0;

}

**基础练习 分解质因数**

问题描述

　　求出区间[a,b]中所有整数的质因数分解。

输入格式

　　输入两个整数a，b。

输出格式

　　每行输出一个数的分解，形如k=a1\*a2\*a3...(a1<=a2<=a3...，k也是从小到大的)(具体可看样例)

样例输入

3 10

样例输出

3=3  
4=2\*2  
5=5  
6=2\*3  
7=7  
8=2\*2\*2  
9=3\*3  
10=2\*5

提示

　　先筛出所有素数，然后再分解。

数据规模和约定

　　2<=a<=b<=10000

本题的C++参考代码如下：

//天农计算机系许晓华老师出品

#include <stdio.h>

#define N 10000//筛法，求1万以内的所有素数

int p[N];

int main()

{

int i,j,a,b,t;

for(i=2;i<N/2;i++)

{

if(p[i]) continue;//合数不参加筛法

for(j=2\*i;j<=N;j+=i)

{

p[j]=1;

}

}

scanf("%d%d",&a,&b);

for(i=a;i<=b;i++)

{

t=i;

printf("%d=",t);

while(t)

{

for(j=2;j\*j<=t;j++)

{

if(!p[j]&&!(t%j))

{

printf("%d\*",j);

t/=j;

break;

}

}

if(j\*j>t)

{

printf("%d",t);

break;

}

}

printf("\n");

}

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

#include<math.h>

int main()

{

long int b,i,k,m,n,w = 0;

scanf("%ld%ld",&m,&n);

for(i = m;i<=n;i++)

{

printf("%ld=",i);

b = i;k = 2;

while(k<=sqrt(i))

{

if(b%k==0)

{

b = b/k;

if(b>1)

{

printf("%ld\*",k);continue;

}

if(b==1) printf("%ld\n",k);

}

k++;

}

if(b>1&&b<i) printf("%ld\n",b);

if(b==i)

{

printf("%d\n",i);w++;

}

}

return 0;

}

**基础练习 字符串对比**

问题描述

　　给定两个仅由大写字母或小写字母组成的字符串(长度介于1到10之间)，它们之间的关系是以下4中情况之一：  
　　1：两个字符串长度不等。比如 Beijing 和 Hebei  
　　2：两个字符串不仅长度相等，而且相应位置上的字符完全一致(区分大小写)，比如 Beijing 和 Beijing  
　　3：两个字符串长度相等，相应位置上的字符仅在不区分大小写的前提下才能达到完全一致（也就是说，它并不满足情况2）。比如 beijing 和 BEIjing  
　　4：两个字符串长度相等，但是即使是不区分大小写也不能使这两个字符串一致。比如 Beijing 和 Nanjing  
　　编程判断输入的两个字符串之间的关系属于这四类中的哪一类，给出所属的类的编号。

输入格式

　　包括两行，每行都是一个字符串

输出格式

　　仅有一个数字，表明这两个字符串的关系编号

样例输入

BEIjing

beiJing

样例输出

3

本题的C++参考代码如下：

#include<iostream>

#include<cstdio>

#include<cstring>

#include<cmath>

using namespace std;

char A[15],B[15];

int main()

{

scanf("%s",A);

scanf("%s",B);

int a=strlen(A);

int b=strlen(B);

int count=0;

if(a!=b) //长度不等

{

printf("1\n");

return 0;

}

//长度相等

else

{

for(int i=0;i<a;i++)

{

if((A[i]!=B[i]))

{

if(abs(A[i]-B[i])!=32)

{

printf("4\n");

return 0;

}

else

{

++count;

continue;

}

}

}

if((count==0))

printf("2\n");

else

printf("3\n");

}

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

#include<string.h>

int main()

{

char a[10],b[10],i,n,l=2;

gets(a);gets(b);

n=strlen(a);

if(strlen(b)!=n)l=1;

else

{

for(i=0;i<n;i++)

if(a[i]==b[i]||a[i]==b[i]+32||a[i]+32==b[i])

if(a[i]!=b[i])l=3;

else ;else {l=4;break;}

}

printf("%d",l);

return 0;

}

**基础练习 字符串对比**

问题描述

　　给定两个仅由大写字母或小写字母组成的字符串(长度介于1到10之间)，它们之间的关系是以下4中情况之一：  
　　1：两个字符串长度不等。比如 Beijing 和 Hebei  
　　2：两个字符串不仅长度相等，而且相应位置上的字符完全一致(区分大小写)，比如 Beijing 和 Beijing  
　　3：两个字符串长度相等，相应位置上的字符仅在不区分大小写的前提下才能达到完全一致（也就是说，它并不满足情况2）。比如 beijing 和 BEIjing  
　　4：两个字符串长度相等，但是即使是不区分大小写也不能使这两个字符串一致。比如 Beijing 和 Nanjing  
　　编程判断输入的两个字符串之间的关系属于这四类中的哪一类，给出所属的类的编号。

输入格式

　　包括两行，每行都是一个字符串

输出格式

　　仅有一个数字，表明这两个字符串的关系编号

样例输入

BEIjing

beiJing

样例输出

3

本题的C++参考代码如下：

#include<iostream>

#include<cstdio>

#include<cstring>

#include<cmath>

using namespace std;

char A[15],B[15];

int main()

{

scanf("%s",A);

scanf("%s",B);

int a=strlen(A);

int b=strlen(B);

int count=0;

if(a!=b) //长度不等

{

printf("1\n");

return 0;

}

//长度相等

else

{

for(int i=0;i<a;i++)

{

if((A[i]!=B[i]))

{

if(abs(A[i]-B[i])!=32)

{

printf("4\n");

return 0;

}

else

{

++count;

continue;

}

}

}

if((count==0))

printf("2\n");

else

printf("3\n");

}

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

#include<string.h>

int main()

{

char a[10],b[10],i,n,l=2;

gets(a);gets(b);

n=strlen(a);

if(strlen(b)!=n)l=1;

else

{

for(i=0;i<n;i++)

if(a[i]==b[i]||a[i]==b[i]+32||a[i]+32==b[i])

if(a[i]!=b[i])l=3;

else ;else {l=4;break;}

}

printf("%d",l);

return 0;

}

**基础练习 时间转换**

问题描述

　　给定一个以秒为单位的时间t，要求用“<H>:<M>:<S>”的格式来表示这个时间。<H>表示时间，<M>表示分钟，而<S>表示秒，它们都是整数且没有前导的“0”。例如，若t=0，则应输出是“0:0:0”；若t=3661，则输出“1:1:1”。

输入格式

　　输入只有一行，是一个整数t（0<=t<=86399）。

输出格式

　　输出只有一行，是以“<H>:<M>:<S>”的格式所表示的时间，不包括引号。

样例输入

0

样例输出

0:0:0

样例输入

5436

样例输出

1:30:36

本题的C++参考代码如下：

//天农计算机系许晓华老师出品

#include <iostream>

using namespace std;

int main()

{

int n,H,M,S,t;

cin>>n;

H=n/3600;

t=n%3600;

M=t/60;

S=t%60;

cout<<H<<":"<<M<<":"<<S;

}

本题的C参考代码如下：

#include<stdio.h>

int main()

{

int a,b,c,d;

scanf("%d",&a);

b=(int)(a/3600);

c=(int)(a%3600)/60;

d=(int)((a%3600)%60);

printf("%d:%d:%d",b,c,d);

return 0;

}

**算法训练 与2，3，5无关的数**

题目背景

　　无聊的题目

出题者

　　Lvat2000

难度

　　EASY

问题描述

　　定义与2，3，5无关的数如下：  
　　（1）位数不得是2，3，5的倍数；  
　　（2）该数不能被2，3，5整除；  
　　（3）该数必须为整数。  
　　写一个判断一个数是否为与2，3，5无关的数的程序，如果有关，输出“No”，如果无关，输出“YeS”。

输入格式

　　一行一个数。

输出格式

　　要求用户的输出满足的格式。  
　　一行，有关输出“No”，无关输出“YeS”.[不含引号]

样例输入

(样例1)13  
(样例2)222  
(样例3)999.9  
(样例4)-7

样例输出

(样例1)No  
(样例2)No  
(样例3)No  
(样例4)YeS

数据规模和约定

　　n<=10^10000;

本题的C++参考代码如下：

#include <bits\stdc++.h>

using namespace std;

int getBits(int n) {

int bits = 0;

while (n != 0) {

++bits;

n = n / 10;

}

if (bits == 0) {

++bits;

}

return bits;

}

int main() {

double eps = 1e-6;

double val = 0.0;

cin >> val;

val = abs(val);

int v = int(val);

if ( abs(val - v) > eps ) {

cout << "No" << endl;

return 0;

}

if (v % 2 == 0 || v % 3 == 0 || v % 5 == 0) {

cout << "No" << endl;

return 0;

}

int bits = getBits(v);

if (bits % 2 == 0 || bits % 3 == 0 || bits % 5 == 0) {

cout << "No" << endl;

return 0;

}

cout << "YeS" << endl;

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

int main()

{

double x;

int y,n;

scanf("%lf",&x);

if((int)x!=x)

{

printf("No");

}

else

{

y=(int)x;

if(y%2==0||y%3==0||y%5==0)

{

printf("No");

}

else

{

for(n=0;y!=0;n++)

{

y=y/10;

}

if(n%2==0||n%3==0||n%5==0)

{

printf("No");

}

else

{

printf("YeS");

}

}

}

return 0;

}

**算法训练 C++ CH08 01**

问题描述

　　已知一个有理数类Zrf\_Ratio，实现如下的操作符重载形式：  
　　friend std::ostream& operator<<(std::ostream&, const zrf\_Ratio&);//输出最简分数  
　　friend std::istream& operator>>(std::istream&, zrf\_Ratio&);  
　　friend bool operator==(const zrf\_Ratio&, const zrf\_Ratio&);  
　　friend bool operator<(const zrf\_Ratio&, const zrf\_Ratio&);

测试

　　测试时主程序会输入四个整数a, b, c, d，表示两个分数a/b和c/d。要求输出最简分数以及两个分数相等和大小的比较结果。

样例输入

1 7 26 25

样例输出

zrf is:1/7; ssh is:26/25  
(zrf==ssh) is:0; (zrf<ssh) is:1

本题的C++参考代码如下：

#include <iostream>

#include <cassert>

using namespace std;

class zrf\_Ratio

{

friend std::ostream& operator<<(std::ostream&, const zrf\_Ratio&);

friend std::istream& operator>>(std::istream&, zrf\_Ratio&);

friend bool operator==(const zrf\_Ratio&, const zrf\_Ratio&);

friend bool operator<(const zrf\_Ratio&, const zrf\_Ratio&);

public:

zrf\_Ratio(int=0,int=1);

zrf\_Ratio(const zrf\_Ratio&);

private:

int num;

int den;

void reduce();//化为最简分数

};

//补充完整构造函数

std::ostream& operator<<(std::ostream& out, const zrf\_Ratio& a)

{

out<<a.num<<"/"<<a.den;

}

bool operator==(const zrf\_Ratio& a, const zrf\_Ratio& b)

{

return a.num\*1.0/a.den==b.num\*1.0/a.den;

}

bool operator<(const zrf\_Ratio& a, const zrf\_Ratio& b)

{

return a.num\*1.0/a.den<b.num\*1.0/a.den;

}

//公有成员函数:

zrf\_Ratio::zrf\_Ratio(int num, int den) : num(num), den(den)

{

reduce();

}

zrf\_Ratio::zrf\_Ratio(const zrf\_Ratio& r) : num(r.num), den(r.den)

{

}

//私有成员函数:

void swap(int &m, int &n)

{

int t;

t=m;

m=n;

n=t;

}

int zrf\_Gcd(int m, int n)

{

if (m<n)

swap(m,n);

assert(n>=0);

while (n>0)

{

int r=m%n;

m = n;

n = r;

}

return m;

}

void zrf\_Ratio::reduce()

{

if (num == 0 || den == 0)

{

num = 0;

den = 1;

return;

}

if (den < 0)

{

den \*= -1;

num \*= -1;

}

if (num == 1)

return;

int sgn = (num<0?-1:1);

int g = zrf\_Gcd(sgn\*num,den);

num /= g;

den /= g;

}

int main()

{

int a = 0, b = 0, c = 0, d = 0;

cin >> a >> b >> c >> d;

zrf\_Ratio zrf(a, b),ssh(c, d);

std::cout<<"zrf is:"<<zrf<<"; ssh is:"<<ssh<<'\n' ;

std::cout<<"(zrf==ssh) is:"<<(zrf==ssh)<<"; (zrf<ssh) is:"<<(zrf<ssh) <<endl;

return 0;

}

本题的C参考代码如下：

该题暂时没有人完全正确，暂时没有该语言的参考程序。

**算法训练 Interval**

问题描述

　　这里写问题描述。

输入格式

　　测试数据的输入一定会满足的格式。

　　例：输入的第一行包含两个整数n, m，分别表示矩阵的行数和列数。接下来n行，每行m个正整数，表示输入的矩阵。

输出格式

　　要求用户的输出满足的格式。

　　例：输出1行，包含一个整数，表示矩阵中所有元素的和。

样例输入

一个满足题目要求的输入范例。

例：

2 2

1 2

3 4

样例输出

与上面的样例输入对应的输出。

例：

10

数据规模和约定

　　输入数据中每一个数的范围。

　　例：0<n,m<100, 0<=矩阵中的每个数<=1000。

本题的C++参考代码如下：

该题暂时没有人完全正确，暂时没有该语言的参考程序。

本题的C参考代码如下：

该题暂时没有人完全正确，暂时没有该语言的参考程序。

**算法训练 最长上升子序列**

问题描述

　　给定一个长为n的序列，求它的最长上升子序列的长度。

输入格式

　　输入第一行包含一个整数n。  
  
　　第二行包含n个整数，为给定的序列。

输出格式

　　输出一个非负整数，表示最长上升子序列的长度。

样例输入

5  
  
1 3 2 5 4

样例输出

3

数据规模和约定

　　0<n<=1000，每个数不超过10^6。

本题的C++参考代码如下：

#include<iostream>

#include<cmath>

using namespace std;

int main()

{

int n;

int \*array;

cin>>n;

array = new int[n];

for(int i = 0; i !=n; ++i)

cin>>array[i];

int \*dp = new int[n];

int number = 0;

for(int i = 0; i != n; ++i){

dp[i] = 1;

for(int j = 0; j < i; ++j){

if(array[j] < array[i])

dp[i] = max(dp[i],dp[j] + 1);

}

if(dp[i] > number)

number = dp[i];

}

cout<<number<<endl;

return 0;

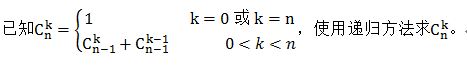
}

本题的C参考代码如下：

该题暂时没有人完全正确，暂时没有该语言的参考程序。

**算法训练 6-1 递归求二项式系数值**

问题描述



样例输入

一个满足题目要求的输入范例。  
3 10

样例输出

与上面的样例输入对应的输出。  
http://lx.lanqiao.cn/RequireFile.do?fid=t7jLy5DD

数据规模和约定

　　输入数据中每一个数的范围。  
　　例：结果在int表示时不会溢出。

本题的C++参考代码如下

#include <iostream>

using namespace std;

int erxianshi(int k,int n)

{if(k==0||k==n)

return 1;

else

return erxianshi(k,n-1)+erxianshi(k-1,n-1);

}

int main()

{ int k,n;

cin>>k>>n;

cout<<erxianshi(k,n);

system("pause");

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

int C(int k,int n)

{

if(k==n)

{

return 1;

}

if(k==0)

{

return 1;

}

return C(k,n-1)+C(k-1,n-1);

}

int main()

{

int k,n;

scanf("%d %d",&k,&n);

printf("%d",C(k,n));

return 0;

}

**算法训练 5-1最小公倍数**

问题描述

　　编写一函数lcm，求两个正整数的最小公倍数。

样例输入

一个满足题目要求的输入范例。  
例：  
  
3 5

样例输出

与上面的样例输入对应的输出。  
例：  


数据规模和约定

　　输入数据中每一个数的范围。  
　　例：两个数都小于65536。

本题的C++参考代码如下：

#include<iostream>

using namespace std;

int main()

{

int a,b,i,j;

cin>>a>>b;

if(a>=b)

{

j=a;

}

else

j=b;

for(i=j;i<=a\*b;i++)

{

if(i%a==0&&i%b==0)

{

cout<<i;

break;

}

}

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

int main()

{

int i=1,j,a,b,n,m,t=1;

scanf("%d%d",&a,&b);

n=(a>b?a:b);m=(a<b?a:b);

while(t%m!=0)

{t=n\*i;i++;}

printf("%d ",t);

return 0;

}

**算法训练 2-3逆序打印数字**

基于例子4 ，写一个程序，在屏幕上打印:

　　9

　　8

　　7

　　6

5

本题的C++参考代码如下：

#include <cstdio>

#include <iostream>

#include <cstring>

using namespace std;

char cMap[4][4];

int n, iMax;

bool Canput(int row, int col)

{

for(int i = row - 1; i >= 0; i --)

{

if(cMap[i][col] == 'o') return false; //判断col列上的合法性

if(cMap[i][col] == 'X') break; //碰到墙，能放，继续判断

}

for(int i = col - 1; i >= 0; i --)

{

if(cMap[row][i] == 'o') return false; //判断row行上的合法性

if(cMap[row][i] == 'X') break;

}

return true;

}

void dfs(int pos, int cur) //对每个单元格pos能否放置碉堡进行判断

{

if(pos == n \* n)

{

if(cur > iMax) //更新最优解

{

iMax = cur;

return;

}

}

else

{

int x = pos / n; //将单元数转换为坐标

int y = pos % n;

if(cMap[x][y] == '.' && Canput(x, y)) //若本单元格可以放置

{

cMap[x][y] = 'o'; //放置一个

dfs(pos + 1, cur + 1); //递归到下一个

cMap[x][y] = '.'; //恢复该单元格，为下一个放置方式作准备

}

dfs(pos + 1, cur); //本单元格不放碉堡

}

}

int main()

{

/\*while(~scanf("%d", &n), n)

{

iMax = 0;

for(int i = 0; i < n; i ++)

scanf("%s", &cMap[i]);

dfs(0, 0);

cout <<iMax <<endl;

}\*/

for (int i = 9; i >= 5; i--) cout << i << endl;

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

int main()

{

int n = 10;

while(--n >= 5){

printf("%d\n", n);

}

return 0;

}